Real-Time Differential Signal Phase Estimation for Space-based Systems Using FPGAs

Shiting (Justin) Lu, Paul Siqueira, *Member IEEE*, Vishwas Vijayendra,
Harikrishnan Chandrikakutty, and Russell Tessier, *Senior Member IEEE*

Abstract

High performance and reliability are important aspects of space-based systems. In many cases, correct system functionality must be continuously monitored to ensure the validity of collected data. In this research we develop a new digital circuit which can detect minute phase differences in time-varying analog signals. These phase shifts can be determined for both a single channel input versus a known model of the signal and across two channels with simultaneously-sampled data. Phase shift data can be used in spaceborne systems to either confirm correct system operation or identify potential problems. Current systems primarily transfer data to the Earth for error checking rather than performing error analysis in real-time. The benefits of our field-programmable gate array (FPGA) based approach are evaluated using a 3 gigasamples per second (GSamp/sec) data acquisition system developed as part of the NASA Surface Water Ocean Topography (SWOT) initiative. Phase calculations with an error of less than 0.021 degrees (0.006% of 360 degrees) are determined using our adaptive approach. The high accuracy of differential phase detection allows for the real-time monitoring of environmental metrics, such as temperature fluctuations, which affect signal phase.

Index Terms

Phase estimation, frequency estimation, amplitude estimation, field programmable gate array, differential phase, real-time, SWOT, GRACE II

I. Introduction

A number of upcoming NASA remote sensing satellite missions will utilize microwave signals to monitor the Earth's environment. A common characteristic of these missions is their reliance on signal phase to characterize acquired data [1]. The National Research Council's decadal survey [2] highlights three such

Shiting (Justin) Lu, Paul Siqueira, Harikrishnan Chandrikakutty and Russell Tessier are with the Department of Electrical and Computer Engineering, Amherst, MA 01003 USA. (e-mail: tessier@ecs.umass.edu).

Vishwas Vijayendra is with Altera Corporation, San Jose, CA, USA.

Manuscript received October 6, 2011, revised March 20, 2012

missions and their importance including: i.) the Tier-1 L-band (1.2 GHz) DESDynI-R mission, which is implementing a spaceborne synthetic aperture radar, ii.) the Tier-2 Ka-band (35.75 GHz) SWOT mission, which is implementing a cross-track interferometer for measuring surface water and ocean topography, and iii.) the Tier-3 GRACE II follow-on mission, which will measure minute changes in the Earth's gravitational field. While each of these missions will utilize signal phase in a different way, the reliance of these missions' science products on signal phase indicates a need for accurate signal phase measurement. As a result, improvements in phase measurement accuracy will have a direct impact on the overall mission architecture and performance.

For the DESDynI-R mission (Deformation, Ecosystem Science and Dynamics of Ice Radar-only), signal phase is used both to form a synthetic aperture which focuses observations into high resolution imagery of the Earth's surface and to analyze a signal's electric field polarization upon return. The received signal's polarization is used to infer structural and orientational characteristics of the scattering target and for the removal of the effects of the ionosphere on the transmitted and received signals. An improved measurement of phase for DESDynI-R would allow for the improved removal of ionospheric artifacts and a better determination of desired target polarization signatures [3].

For the Surface Water and Ocean Topography (SWOT) mission, a signal transmitted by a satellite is reflected from the Earth's surface and received by two antennas on the satellite that are separated by a fixed distance called the baseline (nominally 10m) (Fig. 1). The differential phase of the received signal by the two ends of the baseline is used to infer the angle of arrival of the reflected signal. Since the viewing geometry is known, it is possible to determine the topography of the ocean and inland waters to a high degree of accuracy [4]. The accuracy to which this phase difference can be measured directly affects the intended science product of height. Hence, an improved phase measurement capability can be used to improve overall system performance, or can be traded against other system parameters such as the baseline length, to reduce the size of the spacecraft structure and dramatically impact the overall system cost.

For the GRACE II (Gravity Recovery and Climate Experiment) follow-on mission, which is still in the planning stages, either a microwave or a laser signal will be used to monitor the distance between two co-orbiting satellites. Because satellite orbits are affected by the Earth's gravitational field, a mapping of this changing distance as a function of orbit location can be translated into a map of the gravitational field and used to monitor changes in the field. The very successful GRACE mission [5], which uses 24 GHz and 32 GHz cross-linked signals to accurately monitor the distance between two satellites, has successfully employed the approach. While there are many factors that ultimately contribute to the GRACE mission's acuity in measuring gravitational changes, the measurement accuracy of microwave phase has a direct impact on the final end-product due to its fundamental role in determining the distance between the two satellite platforms.

In all of these systems, it is important to measure a received signal's phase and compare this measurement against a reference signal (as in the GRACE II system), or a signal received by another antenna (as in

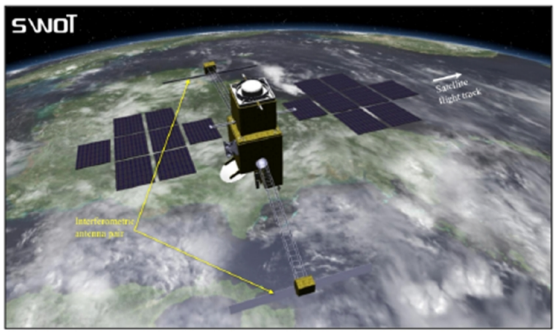

Fig. 1. An illustration of the planned SWOT satellite mission. An interferometric pair of satellite antennas are located at the top and bottom of the figure. Signal phase differences from the antennas are used to examine the topography of bodies of water such as oceans, rivers, and lakes.

SWOT). The characterization of a system's phase measurement performance may occur in the final stages of spacecraft assembly or as part of routine on-orbit operations. A typical phase characterization may involve the injection of a known signal into the nominal science data path, the recording of the characteristics of the known signal, and a comparison against a standard.

In the time period before launch, measurements are made to assure that overall satellite operation is able to meet engineering specifications. In the pre-launch period, access to key internal workings of the satellite becomes more challenging as the system achieves higher levels of integration. Generally, this configuration more closely mimics what will ultimately be deployed in space. As such, during pre-launch, access to the large volumes of data and intermediate products necessary to accurately characterize satellite performance becomes difficult and it is more desirable to measure the science end-product directly on the satellite system.

After launch, it is often necessary to monitor overall system performance on a per-pulse or per-orbit basis, throughout the lifetime of the mission. Due to data rate limitations, it is desirable to perform corrections on-board, as necessary, to eliminate the need for transmitting calibration data intermediate products to the ground. If these health-checks can be performed automatically, and only upon failure to notify the ground that a problem has been detected, and then transmit additional data as necessary, a significant savings in terms of manpower and data rate can be achieved while still gaining in system robustness.

For these reasons, and the importance of performing high-fidelity phase measurement on-board satellites, it is desirable to implement low computational overhead phase stability calculations directly in a mission's science-data signal path. Information from these calculations can then be used to monitor the overall health of the satellite in real time and provide the ability to continuously monitor system health at time scales commensurate with those of the raw science data, which may be on the order of milliseconds to seconds.

FPGA-based data processing solutions have been widely implemented due to their performance, flex-

ibility, and ability to update system functionality [6] [7] [8]. In particular, these systems allow for high-performance on-board processing which can meet speed and robustness requirements for space applications. One such FPGA-based example is the Venus Express Monitoring Camera VMC [9]. The configurability of FPGAs makes the system adaptive to different applications and brings good flexibility [10] [11] [12]. In this paper, we present an adaptive implementation of a hardware circuit which allows for the continuous monitoring of signal phase. This implementation is migrated from a phase monitoring software algorithm [1] that has previously been shown to provide accuracies that achieve theoretial bounds set by the number of samples collected and errors associated with the input data.

The new hardware circuit has been directly implemented into a field-programmable gate array (FPGA) that is interfaced to two high-speed analog-to-digital converters (ADCs). While the presented algorithm can be implemented on any sufficiently sized FPGA, this particular FPGA-based system has been designed to perform computation in the signal path for the SWOT mission. By measuring and monitoring the differential phase of a known signal fed into the two analog-to-digital converters, a low-level check can be performed on the science data integrity on a pulse-to-pulse basis. The results are subsequently passed down and incorporated into a standard data telemetry data stream. On-board FPGA monitoring of these results can be used as a "watch dog" to flag problematic data and to record intermediate products that can be saved and sent to the ground for further analysis. By utilizing an adaptive component such as an FPGA for this process, the system can be configured over the lifetime of the satellite to alter the threshold at which detailed and data intensive analysis takes place. As a result, the platform provides an efficient and insightful approach for monitoring overall satellite system performance.

Through experimentation using a block of 2400 samples with 30dB SNR, we show that the FPGA circuit required to perform phase monitoring easily fits within FPGA resources of our board (about 20,000 logic blocks) and can achieve high accuracy (the error is less than 0.021 degrees which is 0.006% of 360 degrees). Our approach is demonstrated in real-time using a signal generator and an FPGA-based system. Experimentally, we show that real-time differential phase changes can detect changes in environmental factors, such as temperature. Since temperature effects occur at time scales on the order of seconds, real-time in this context is meant to indicate calculations being performed at this scale or better, and able to keep up with a large sampling data rate from the ADCs.

The rest of this paper is organized as follows. Section II provides background on the need for real-time phase detection and its possible use in the SWOT system. In Section III, our hardware-implemented phase detection algorithm is described in detail. In Section IV, the implementation of the FPGA circuit used for processing is described. The experimental approach used to generate results is presented in Section V, while the results themselves are provided in Section VI. Section VII concludes the paper.

II. Background

Over the past few years, the University of Massachusetts has developed a dual-channel Ka-band (35 GHz) receiver with an integrated digital processing platform (Fig. 2) [13]. This integrated system forms the

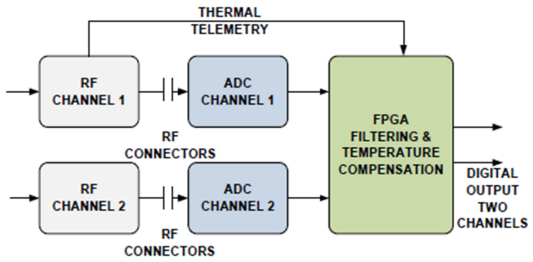

Fig. 2. An illustration showing integrated RF and digital components for SWOT. Two 35 GHz input signals are downconverted to 1.2GHz before being sampled by dual 3 GSamp/sec ADCs. Sampled results are then input to a Xilinx XC4VFX140 FPGA. Circuitry implemented inside the FPGA performs additional digital downconversion and signal compensation to address thermal fluctuations in the receiver RF front-end.

basis and motivation for our phase error detection work. The digital system implementation includes dual National Semiconductor ADC08D1520 3 GSamp/sec analog-to-digital converter chips (ADCs), a Xilinx XC4VFX140 field-programmable device, and a variety of external interfaces, including a compact PCI bus port. The digital system's front-end (ADCs and FPGA) is designed to downconvert a 200 MHz bandwidth signal centered at 1.3 GHz to one centered at 150 MHz. Here, for testing purposes, this same system is configured to downconvert a 1.2 GHz signal to 62 MHz.

The combination of the receiver RF and digital system is able to address temperature-induced fluctuations in signal amplitude and phase created by the RF receiver front-end. Since the RF receiver operates at high frequency (35.75 GHz, a wavelength (λ) of 8.4 mm), even minute temperature fluctuations between the two analog input channels will cause variations in the associated electrical signal path lengths. In this instance, phase differences that are not related to the measured topographic height (e.g. Fig. 3) will appear, possibly leading to significantly-increased system error.

To address the thermally-induced phase error issue, a complete RF and digital receive system has been designed and constructed in the laboratory [15]. This system identifies the accuracy of the RF front-end's performance relative to temperature and then adjusts sampled and downconverted results in real-time to rectify the effects of temperature-induced signal fluctuations. These signal corrections take place using digital circuitry implemented in FPGA logic and storage resources. Corrections of both input amplitude and phase within the 200 MHz signal passband are required. Not only is the phase variation of a single channel of interest, but the relative phase difference between the two downconverted input channels is also a concern. Following an assessment of differential phase during data collection, remediation efforts must be performed to address changes in the electrical path lengths of the two input RF channels. The real-time processing of the sampled data necessitates the optimization of our phase detection approach for compute speed and FPGA resource efficiency. During system testing, it has been shown that the digital board with two ADCs and a Xilinx FPGA can process a 4.4 kHz pulse repetition frequency (PRF) and 30

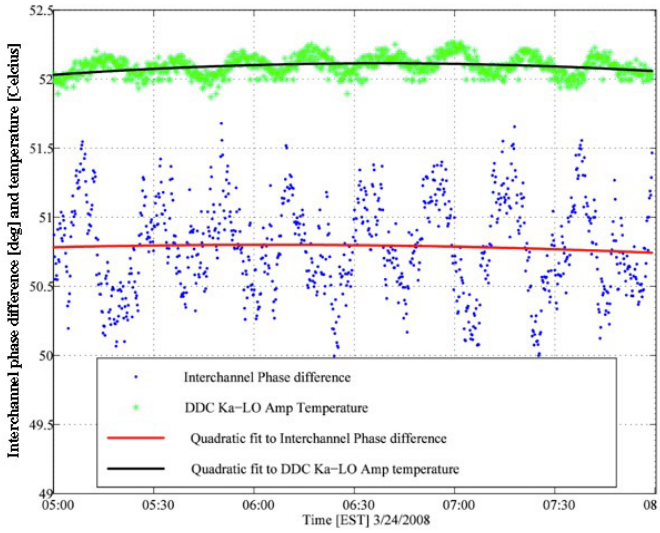

Fig. 3. The relationship between receiver thermal temperature (green) and the differential phase of two downconverted signals (blue). The periodic temperature variation caused by the testing environment is apparent from the differential phase plot. Data for the graph was first presented in [14]. Both temperature and differential phase are represented in quantities labeled on the left vertical axis.

sec sampling window for both channels sampling at the full rate of the ADCs (3 GSamp/sec).

In previous works [1] [13], a software algorithm was shown to provide measurements of signal phase whose accuracy could be predicted and/or controlled by the numbers of samples and the signal-to-noise ratio. The algorithm operates on a single-tone signal which has been sampled using an analog-to-digital converter. A comparison is made between the waveform determined through sampling and the expected single-tone signal in terms of frequency and phase. Observed variations between the reference and sampled waveform indicate the gain and phase characteristics of the input signal's path through the system's RF front-end. The phase and gain accuracy of these values is set by the quantity of sampled data and the signal's signal-to-noise ratio (SNR). An increase in either the size of the sample set or the SNR value improves the accuracy of phase and gain estimates by an amount which can be determined mathematically. The approach used to estimate these values can then be tuned to acquire a suitable accuracy level for their measurement.

Plots of measured and predicted phase accuracy versus an increasing number of samples are shown in Fig. 4. Predicted and measured phase accuracy, shown as a standard deviation versus sample count is presented using a logarithmic scale. In this plot, the number 10 represents 2^{10} samples, or 1024 samples. Matlab simulations using 5, 10, and 20 dB SNRs and sample counts between 16 (2^4) and 4,096 (2^{12}) were performed. The figure illustrates a match between the values determined via a numerical estimation routine using Matlab (sampled points) and the theoretical accuracy which assumes a statistically optimal estimate of the signal phase for a given number of samples and signal-to-noise ratio. The excellent agreement

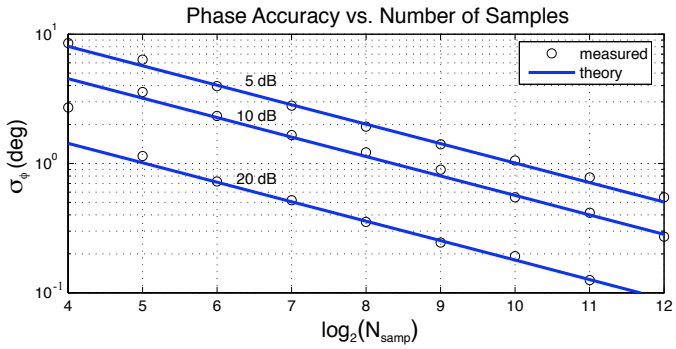

Fig. 4. Measured and theoretical phase accuracy versus the number of samples. As sample count increases, the accuracy of the measured phase values is improved [1].

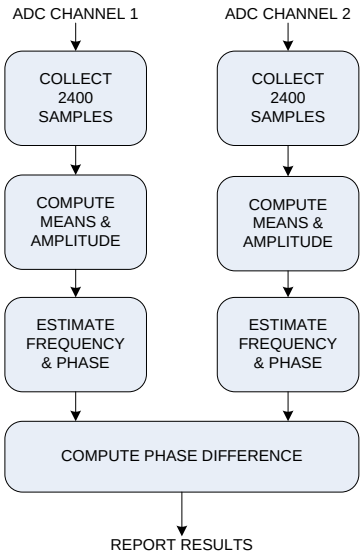

Fig. 5. Data flow of the hardware algorithm needed for FPGA differential phase estimation. Although 2,400 samples are used per phase calculation in our experimentation, this number can be increased if increased estimate accuracy is required or the signal-to-noise ratio of the input data is decreased.

between theory and simulation indicates that the algorithm is maximally efficient in terms of the number of data points it collects versus the accuracy of the resulting estimates.

A hardware implementation of the algorithm (Fig. 5) thus allows for the real-time analysis of the sampled RF data with optimal accuracy for the given input data. In a nominal implementation, a known single-tone signal can be periodically passed through the system and compared to expected results. These calculated real-time phase and amplitude values form a data set that is orders of magnitude smaller than the sampled data set of the original waveform. If hardware-calculated phase and amplitude values are found to deviate from expected values, sample data originating from the system can be isolated from the science data stream and a detailed system diagnostic can be performed.

The work described in this paper greatly extends our previous work in hardware-based phase estimation [16], which itself is based on the algorithm first described in [1]. Due to a limited accuracy FPGA-based frequency estimation approach [16], the phase estimation for a single channel signal resulted in up to a 3.1 degree estimation error (0.9% of 360 degrees). By analyzing the correlation between frequency error and phase error in the original algorithm, we found that the estimation error of phase linearly increases with the frequency error for a fixed block length. In this paper, frequency is first estimated by counting the number of cycles in a waveform over a fixed period of time, as in [16]. The block of data is then divided into sub-blocks and the phases of each sub-block is estimated using the previously calculated frequency. A corrected frequency is calculated by applying a least squares estimator for calculating the slope of a line between the time axis and sub-block phase offsets. The final, corrected phase is obtained through iterations of the phase estimation technique with this corrected frequency. This estimation approach is shown to be more than two orders of magnitude more accurate for this work compared to the previous one due to the use of a more accurate frequency estimate. With this improved accuracy, a series of differential phase detection experiments are conducted to demonstrate the application of the phase estimation hardware. Particularly, differential phase measurements using cables that have been subjected to heating and cooling are performed to demonstrate the sensitivity of our approach.

III. FPGA-Based Phase Calculation Algorithm

The phase detection algorithm is block based. A block of data contains M consecutive samples. The phase of the first sample in the block is called the initial phase. Samples are stored in an array called $v[]$, which represents the signal's measured *voltage*. For this implementation, it is assumed that the size of the array, M, is greater than 1000 and the array covers multiple cycles of the input sinusoidal waveform. Operations take place in the following sequence:

A. Determine and Count Signal Zero Transitions

The FPGA-based hardware algorithm begins by fetching a total of M samples from array $v[]$. Each sequence of three consecutive $v[]$ samples are then *block*-averaged using a three-point boxcar filter to create a new sequence of values $vb[j] = (v[j-1] + v[j] + v[j+1])/3$, to reduce digital and thermal noise. Here, $j = 1, 2, ...M$. The generated block-averaged voltage values, $vb[]$, are used to find negative-to-positive transitions in the sinusoidal data stream. These negative-to-positive transitions are called zero transitions in later sections. The number of these zero transitions (N_{zero_v}) in the $vb[]$ array is determined along with the index in $vb[]$ where the first ($jvz[1]$) and last ($jvz[N_{zero\_v}]$) zero transitions in the data stream occur.

B. Determine the Sample's Sum of Squares

Samples in $v[]$ are also used to calculate the sum of squares of sample values for all points such that $C2[j] = \sum_{i=1}^{j} v[i]^2$, where the variable $C2$ is meant to indicate the *cumulative* sum of the squares. To

maintain computational efficiency, the intermediate values $vb[]$, N_{zero_v}, $jvz[1]$, $jvz[N_{zero\_v}]$ and $C2[]$ are calculated in parallel in the FPGA. These intermediate values are used to calculate the estimated amplitude, frequency, and phase of the signal represented by $v[]$.

C. Estimated Amplitude Calculation

The input signal is modeled as

$$s(t) = A\cos(2\pi f_0 t + \phi_0). \tag{1}$$

where A is the signal amplitude, f_0 is the signal frequency, and ϕ_0 is the signal phase.

The amplitude, A, can be determined by measuring the energy of the signal, which is proportional to the square of the voltage. The integral of $s^2(t)$ in one cycle is given by

$$\int_0^{T_s} s^2(t)dt = \frac{A^2 T_s}{2}. \tag{2}$$

Here, T_s is the period of the input sinusoidal signal which also equals $1/f_0$. Using this relationship, it is possible to calculate the energy of the signal during the time period from the first sampling point to the $jv[N_{zero\_v}]^{th}$ sampling point. The energy can be obtained by multiplying the energy in one cycle given by (2) with the number of cycles during this period. Alternatively, the energy can be determined from measurements by multiplying $C2[jvz[N_{zero\_v}]]$ by the sampling interval, T_{samp}. This leads to the relationship

$$\frac{A^2 T_s}{2} \times \frac{jvz[N_{zero\_v}] \times T_{samp}}{T_s} = C2[jvz[N_{zero\_v}]] \times T_{samp} \tag{3}$$

In the above equation, the multiplier on the left side is the number of signal cycles, determined by dividing the duration of the time period by T_s. After simplification, signal amplitude can be calculated by

$$A = \sqrt{\frac{2}{jvz[N_{zero\_v}]}} \times \sqrt{C2[jvz[N_{zero\_v}]]} \tag{4}$$

which is easy to compute from the FPGA's determination of intermediate values. The equation benefits from the integration of the signal over multiple signal cycles, as opposed to calculation using just one signal cycle.

The numerator in the expression includes the sum of sample squares of all points up to the index of the last data zero transition. The denominator is the index of the last data zero transition of the sinusoidal signal. A square root is taken of the resulting value to determine the amplitude value.

D. Initial Frequency Estimate

There are $N_{zero\_v} - 1$ cycles of the input signal from the first to the last zero transition in one block. The number of samples during this time period is $jvz[N_{zero\_v}] - jvz[1]$. Since the average number of samples in one cycle and the sampling frequency are known from these calculations, the signal frequency can be determined. The estimated frequency of the signal is determined using:

$$f = \frac{N_{zero\_v} - 1}{jvz[N_{zero\_v}] - jvz[1]} \times f_{samp} \tag{5}$$

where, f_{samp} is the system sampling frequency of the analog-to-digital converter. Since the number of samples during $N_{zero\_v}-1$ cycles is not necessarily consistent, error may be introduced using this method. It is critical to accurately estimate the frequency to get the required accuracy for phase estimation. From the results reported in [16], the phase error of this method is determined to be about 2%. An estimate refinement method which provides a much improved frequency estimation from this initial estimate is given in Section III-G.

E. Initial Estimate of Signal Phase

The phase of a block of data can initially be determined using

$$\phi_{init} = 2\pi N_{zero\_v} - \frac{\pi}{2} - \frac{2\pi f}{f_{samp}} \times jvz[N_{zero\_v}] \tag{6}$$

where the third term on the right represents the phase change from the initial point to the last zero transition point.

F. Improved Phase Estimate Using an Iterative Method

To improve the phase calculation accuracy, an iterative process is used to obtain more precise estimation. The following iterative formula is based on the work described in [1] to estimate signal phase. The updated phase for the current estimate, ϕ_c^k is based on the phase from the previous estimate, ϕ_c^{k-1}, the recorded voltage waveform, $v[]$ and the trigonometric function of $\sin$ and $\cos$, as in:

$$\phi_c^k = \phi_c^{k-1} - \frac{\sum_{m=1}^{M}(\sin\theta_m \times (v[m] - A\cos\theta_m))}{A\sum_{m=1}^{M}(\sin\theta_m)^2} \tag{7}$$

where $\omega = 2\pi f, \Delta t = 1/f_{samp}$, and

$$\theta_m = \omega \times \Delta t \times (m-1) + \phi_c^{k-1}. \tag{8}$$

In the first iteration, ϕ_c^0 equals ϕ_{init} from (6). The corrected initial phase is effectively refined by evaluating estimated phase values across all M data points in $v[]$. From our experimentation, a total of 3 iterations are adequate to approximate the final converged value.

G. Corrected Frequency Estimate

As mentioned in Section III-D, the frequency determined from (5) is not sufficiently accurate to achieve phase calculations that reflect the theoretical accuracy limits demonstrated in Figure 4. Frequency estimation can be improved from the initial estimate in (5) by analyzing the progression of estimated phase as a function of time. To demonstrate this process, it is first shown that the difference between the measured phase and estimated phase (herein, the *phase error*) is linear with frequency error. Then, an estimation method based on this property is described which corrects the estimated frequency. In this method, a data block under analysis is divided into sub-blocks and the frequency obtained from (5) is used to calculate phases for the sub-blocks. Although these sub-block phases are less accurate than than one obtained if

the entire block is used, their progression as a function of time can be used to calculate the corrected frequency.

First, the relationship between frequency estimation error and phase error is shown. The value Δf is the frequency error $\Delta f = f - f_0$ where f_0 is the actual frequency of the single tone input signal and f is the one estimated by (5).

Ideally, the iteration stopping condition for (7) could be set to when the error term in (7) becomes zero. Thus,

$$\sum_{m=1}^{M} \sin\theta_m (v[m] - A\cos\theta_m)\Delta t = 0 \tag{9}$$

If the sampling frequency is high enough that the left term of (9) is approximately equal to its integral representation, then

$$\begin{aligned}
&\int_0^T \sin(2\pi f t + \phi)[A\cos(2\pi f_0 t + \phi_0) - A\cos(2\pi f t + \phi)]dt \\
&= -\frac{A}{4\pi(f+f_0)}\cos[2\pi(f+f_0)t + (\phi+\phi_0)]|_0^T && (10)\\
&\quad -\frac{A}{4\pi\Delta f}\cos(2\pi\Delta f t + \Delta\phi)|_0^T && (11)\\
&\quad +\frac{A}{8\pi f} cos(4\pi f t + 2\phi)|_0^T = 0 && (12)
\end{aligned}$$

In the above, ϕ and ϕ_0 are the estimated and actual phase, respectively, $\Delta\phi = \phi - \phi_0$ and T is the time duration of the block, $T = M\Delta t$, which is constant under a fixed sample rate and a fixed block length. Since f_0 and f are much larger than Δf, the second term (11) is dominant in summation and the first term (10) and the third term (12) can be ignored.

Hence, after evaluating at the limits, and following (8) we have

$$\begin{aligned}
&\cos(2\pi T\Delta f + \Delta\phi) - \cos(\Delta\phi) \\
&\qquad = -2\sin(\pi T\Delta f + \Delta\phi)\sin(\pi T\Delta f) = 0,
\end{aligned} \tag{13}$$

from which it can be shown that

$$\Delta\phi = -\pi T\Delta f = -\pi M\Delta t\Delta f. \tag{14}$$

Thus, the link between the phase error and frequency error has been established.

The relationship given by (14) can be used to calculate the corrected frequency. The original block of data is divided into several sub-blocks of equal length, N_{sub}, and the phase for each sub-block is determined independently using the iterative method described in Section III-F. The frequency and amplitude determined in Sections III-D and III-E are used in the iteration. The initial phase used as iteration input for each sub-block is obtained by shifting $2\pi f\Delta t N_{sub}$ from the sub-block which precedes it. The final phase error for each sub-block after iteration can be assessed using (14), where the variable M is replaced in the equation with N_{sub}, the sub-block length used in iteration. Since all sub-blocks have

the same block length, the frequency error introduces a constant error in the phase of each sub-block, as determined by (14) and illustrated in Fig 6.

As an example, the phase and time pairs for four sub-blocks of 600 samples each are plotted in Fig. 6. The two dashed lines are the measured phase-time curves before and after iteration to determine the individual sub-block phases. These lines have different slopes because prior to iteration the frequency error propagates throughout the phase estimates. After iteration, the phase, and hence the frequency, is better estimated. The solid line in Fig. 6 is used to depict the actual phase curve. After the phase estimate is calculated through iteration for each sub-block, the phase at the beginning of each sub-block has a constant offset with respect to the actual phase at each point. In this example implementation, the number of samples in each block is 2400 and the block is divided into 4 sub-blocks of 600 samples each. The measured phase vector after sub-block iteration is denoted as $Y = [\phi_{1c}, \phi_{2c}, \phi_{3c}, \phi_{4c}]$. Here, ϕ_{ic} are the measured phases for the sub-blocks. Letting the time vector be $X = [0, 600\Delta t, 1200\Delta t, 1800\Delta t]$, the slope of the phase curve after sub-block iteration can be obtained using linear fitting, which makes the sum of point-line distances the smallest (least squares method). In this case, the least squares estimate for the slope of a line is given by

$$\text{slope} = \frac{\frac{1}{4}\sum X_i Y_i - (\frac{1}{4}\sum X_i)(\frac{1}{4}\sum Y_i)}{\frac{1}{4}\sum X_i^2 - (\frac{1}{4}\sum X_i)^2} \tag{15}$$

Here, X_i and Y_i are elements of vectors X and Y noted above and the slope is equal to $2\pi f_c$, as seen in Fig. 6. Further simplification by plugging values into (15) leads to

$$\omega = 2\pi f_c = \frac{-3\phi_{1c} - \phi_{2c} + \phi_{3c} + 3\phi_{4c}}{6000\Delta t} \tag{16}$$

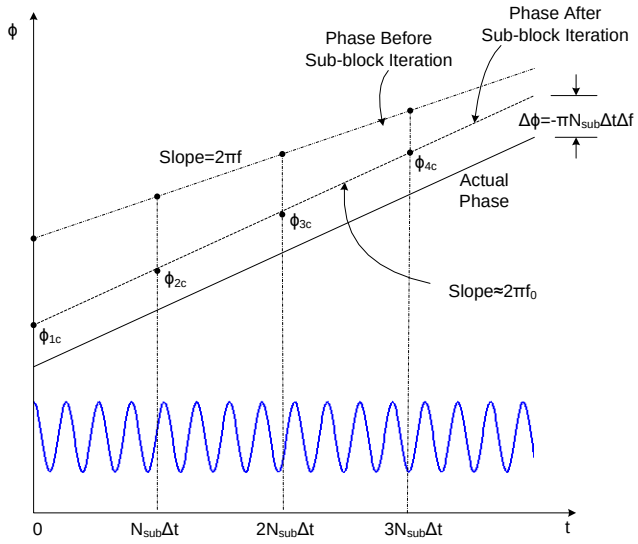

Fig. 6. Illustration of the sub-block method for calculating frequency. The lower part of the plot illustrates a voltage signal as a function of time which is broken up here into four sub-blocks. Phase curves are in the upper part.

for a block of 2400 samples split into four equally-sized blocks of 600 samples each. As a result, the corrected frequency which is determined from (16) is more accurate than the estimate determined in Section III-D. By using this frequency in (7) and (8), the required accuracy for the final phase estimation is obtained.

H. Final Iteration Using The Corrected Frequency

Once the corrected frequency is obtained using the approach of Section III-G, it can be used iteratively in the phase estimation process described in Section III-F. Unlike the calculation to determine the initial phase for the sub-blocks, the whole block is used in this iteration. Our experimentation shows that, generally, three iterations of frequency and initial phase estimation are enough to converge to a phase estimate that does not change with more iterations.

IV. FPGA Implementation of Phase Detection Algorithm

The phase detection algorithm described in Section III was implemented in a Xilinx XC4VFX140 Virtex-4 FPGA. A diagram of the phase calculation implementation is shown in Fig. 7(a). The ADC has a 1.5GHz sampling clock and data are double-edge sampled. Sample data are transferred from the ADCs to the FPGA as four separate 375 MHz data streams (8-bit width) operating at a dual data rate (DDR). The 375 MHz digital clock (the original 1.5 GHz clock divided by four) is fed into the FPGA from the ADC for use in data capturing. The clock is further divided by four within the FPGA to generate a 93.75MHz clock. Input data values are demultiplexed by 1:8 ISERDES circuitry inside the FPGA using the 375 MHz clock. Individual data words are reconstructed using the SCRAMBLER block shown in Fig. 7 and input into a 256x1024 FIFO. The 93.75MHz clock is used as the write clock for the FIFO.

A. Phase Calculation Hardware

For differential phase detection, two ADCs, IADC and QADC, are used and the phase calculation hardware in Fig. 7(a) is duplicated for each channel. One source clock is split and transmitted to the ADCs through separate, half-meter cables. Although the IADC and QADC have the same input clock source, two distinct 375 MHz digital output clocks are sent from the ADCs to the FPGA, one for each channel. This creates two synchronization issues for two channel ADC sampling. We ensure that the 375 MHz digital output clocks have the same phase by using a single synchronization signal from the FPGA to align the clocks in both ADCs. When the synchronization signal is asserted, the two ADCs align their output clock signals using circuitry built into the ADCs. The other issue results from the possible slight skew of internal FPGA clock routing which may make the FIFO write clocks for I and Q channels plesiochronous. Since the write enable signal for the FIFOs is asserted asynchronously to these clocks, the signal must be synchronized to the 93.75MHz clock domain before it is applied to the FIFO write enable inputs. This action allows the FIFOs to receive data sampled from the same ADC source clock rising clock edge. Two

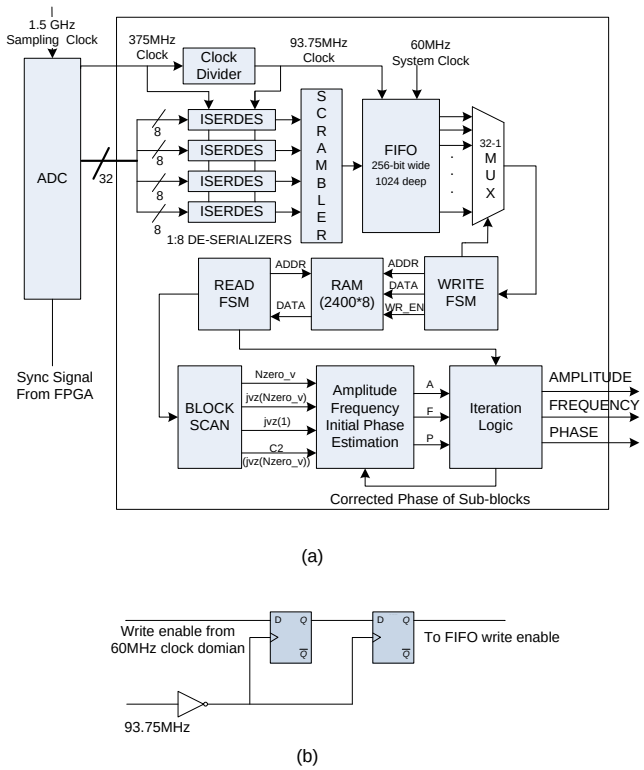

Fig. 7. a.) Block diagram of the phase calculation logic used in the Data FPGA to implement the algorithm that estimated amplitude, frequency, and phase of the reference cosine signal. b.) Synchronization flip-flops for FIFO write enable.

flip-flop chains (one for each channel) are used to synchronize the write enable signal, as shown in Fig. 7(b). The input clock to the synchronization registers are inverted (use negative edge triggered registers) to guarantee that the FIFOs in the two channels begin to store data at the same subsequent rising clock edge.

The RAM in Fig. 7 contains one 2400 sample block of 8-bit data. For each block, a state machine fetches data samples from the FIFO, performs data format conversion, and stores them in the RAM. As values are collected from the FIFO they are converted from 8-bit unsigned values to two's complement representation by the subtraction of 127. The array values stored in RAM are located in adjacent memory locations. The block scan module reads the sample block and generates the zero transition point number (N_{zero_v}), the sample indices for the first and last zero-crossing points ($jvz[1]$ and $jvz[N_{zero\_v}]$) and the accumulation of squares of sample values ($C2[jvz[N_{zero\_v}]]$) for the next stage calculation.

The Amplitude, Frequency and Initial Phase estimation module implements most of the algorithm described in Section III, except for iteration control. Since data accuracy is needed for iteration, floating point units are used in this module. The floating point units are generated by the CORE generator from Xilinx [17]. The Iteration Logic module implements (7) in Section III-F to determine the corrected phase for both sub-blocks and the entire block. The module requires the use of 32-bit Xilinx CORDIC blocks

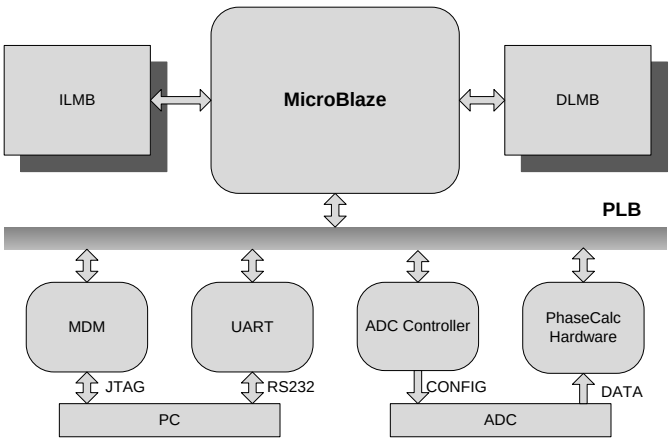

Fig. 8. System diagram of the data analysis FPGA logic including a MicroBlaze processor.

to perform sin and cos operations [18].

B. Software and Hardware Integration

The overall system architecture of the circuitry inside the FPGA is shown in Fig. 8. The system is composed of a MicroBlaze microprocessor, instruction (ILMB) and data (DLMB) RAMs and some peripherals. The processor local bus (PLB) is used to interconnect the processor and peripherals. Software and hardware are used collaboratively to determine the differential phase. The software executed in the Micro-Blaze is responsible for ADC configuration, ISERDES and FIFO enabling, phase calculation logic reset and results collection.

A UART is used for logging system information and transmitting results to an attached PC. The ADC controller module in Fig. 8 configures the ADC control registers after receiving information from the MicroBlaze. The PhaseCalc module implements the circuitry shown in Fig. 7. The MicroBlaze accesses the results of this module via programmable registers.

V. Experimental Approach

A hardware implementation of the phase detection algorithm from Section IV was tested using the system illustrated in Fig. 9. A 62 MHz single-tone signal is generated by an arbitrary waveform generator and used as a reference. A splitter is used to divide the input signal into two equivalent signals. A separate splitter is used to divide a 1.5 GHz sinusoidal clock signal into two signals which are used as ADC clocks. The half-meter cables from the clock splitter have the same length and are located close to one another, so they can be at the same temperature. The slight skew between the two clocks caused by cable imperfections is small enough that it does not impact the required accuracy of our algorithms. Sampling is performed on both clock edges, allowing for a 3 GSamp/sec sample rate. The heart of the system is a new custom-designed data acquisition system built to process L-band signals for the SWOT project. This

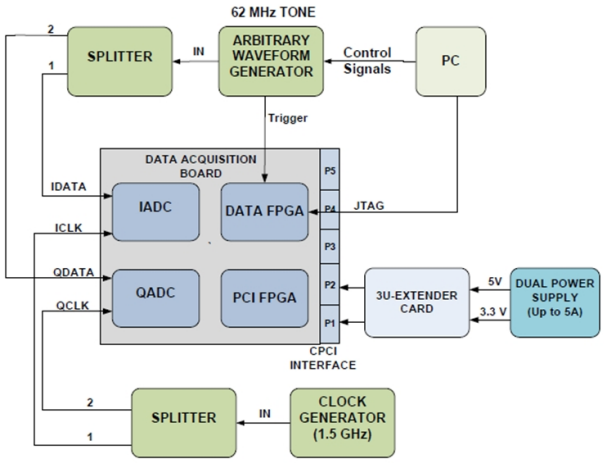

Fig. 9. Illustration of experimental setup for the phase detection circuit implemented in the laboratory

FPGA-based system (Fig. 10) includes 2 National Semiconductor ADC08D1520 3 GSamp/sec analog-to-digital converters and a Xilinx XC4VFX140 Virtex-4 FPGA (labeled Data FPGA in the figure). Although a UART was used for data transfer in this work, data can be streamed off the board using SATA and SFP connectors which handle 1 Gbps data rates from Xilinx RocketIO Multi Gigabit Transceivers (MGT).

A set of steps was followed to collect data for our experiments:

1) The 1.5 GHz clock is enabled to start ADC sampling
2) The 62 MHz input signal is enabled in the waveform generator
3) A start trigger is sent from the signal generator to the FPGA
4) This trigger enables the FIFOs to collect and store data
5) Processing starts once the input FIFO is full

An amplitude, frequency, and phase estimation circuit based on the algorithms outlined in Section III was implemented in the Data FPGA. This information is used by the FPGA to determine if the RF system is operating within expected parameters. During space-based operation, diagnostic and science data would be transferred to a terrestrial base for enhanced examination. For these experiments in the lab, sampled data and amplitude, frequency, and phase information are transferred to a desktop computer using a 56 Kb/sec serial cable.

VI. Results

A. Resource Utilization and Performance Evaluation

The Data FPGA includes two types of estimators for the two signals feeding the data acquisition system. Table I shows the post-synthesis hardware utilization of single- and dual-channel estimators implemented in the Data FPGA. The single-channel estimator is used for a single input channel only. Two single-channel

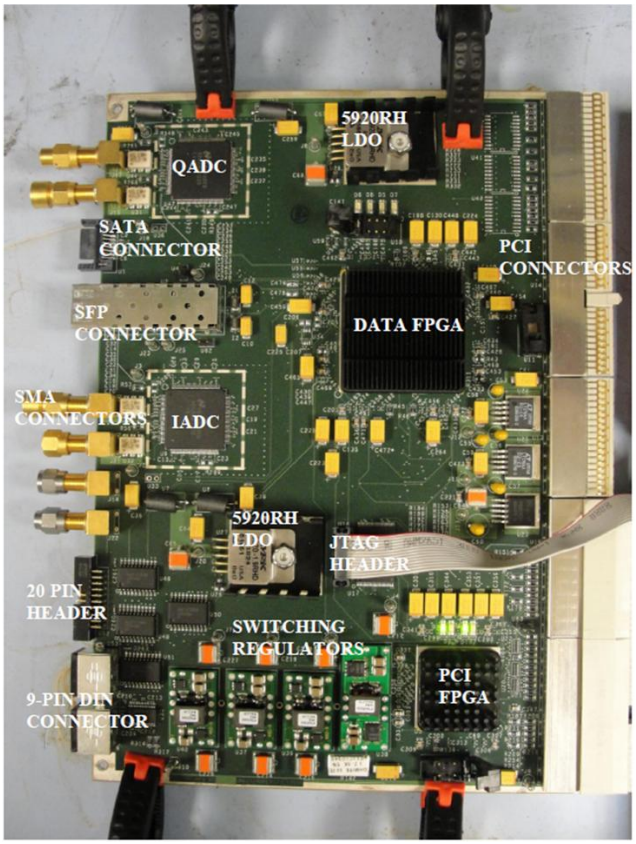

Fig. 10. The data acquisition board constructed as a SWOT prototype. The board contains two 3 GSamp/sec ADCs and a Virtex-4 FPGA.

estimators are needed to estimate phase for both channels simultaneously. The dual-channel estimator is used for differential channel phase for the two channels.

To evaluate phase calculation performance, an initial experiment was performed for a 62 MHz input signal for the Q channel ADC (QADC). The clock speed of the FPGA circuitry was set to 60 MHz. A total of 10.4 ms was needed to determine the frequency, amplitude and initial phase of the input signal and 10.3 ms was required to determine the final phase estimate. These execution times were determined

TABLE I

HARDWARE UTILIZATION OF SINGLE- AND DUAL-CHANNEL ESTIMATORS IMPLEMENTED IN THE DATA FPGA

FPGA Logic Resources	**Single Channel**	**Dual Channel**
LUTs	22,693	37,175
Flip Flops	8,732	27,175
RAM Blocks	33	66
DSP Blocks	39	73

starting from the initial sample read from RAM. Overall, less than 200 ms were needed to calculate phases for 8 blocks, and approximately 48 blocks can be processed in one second.

The same algorithm was implemented in C code and executed on the MicroBlaze at 60 MHz, the same clock speed as the hardware implementation. We measured the execution time of the phase calculation by reading a clock cycle counter attached to the PLB bus. For single block (2400 samples) processing, the software implementation takes 1.57 seconds if on-chip block RAM is used for both instruction and data memory. A total of 3.46 seconds are required to finish the same workload if external DDR memory is used for data memory (mainly the program stack). By comparison, the FPGA implementation takes 20.7 ms for the same computation ($10.3 + 10.4$ ms for single block processing). Overall, this hardware implementation outperforms the MicroBlaze-based software implementation by about two orders of magnitude.

The experimental results obtained from using the FPGA implementation match our operational real time constraints of phase error evaluation each second. The 1.57 second analysis time required by the MicroBlaze implementation is insufficient to meet these requirements.

B. Single Channel Phase

The first set of our experimental measurements consisted of using a simple 62-MHz sinusoidal signal to measure initial phase for each block stored in the on-board FPGA FIFO. A shadow FIFO which contains exactly the same sample data is used to dump samples to a PC for post-processing to validate the results. Because the data in the FIFO comes from continuous sampling, the phase offset between two consecutive blocks is given by

$$\phi_{block_i} = \phi_{block_{i-1}} + 2\pi T \times f_0. \tag{17}$$

As described in Section III-H, three iterations were necessary to converge to the results in Table II. The input signal's true initial phase was determined in post-processing using a 62 MHz golden frequency, 32,768 samples, and double precision arithmetic. For comparison, the initial phases for all blocks except sample block 1 were adjusted using (17).

The largest estimated error is 0.021 degrees. The theoretical lower bound for the error in measuring signal phase [1] is

$$\sigma_\phi^2 = \frac{1}{N}\frac{1}{R_{SNR}} \ \text{rad}^2 \tag{18}$$

where

$$R_{SNR} \approx 6q + 1.8 \text{ dB}. \tag{19}$$

The variable q is the effective number of bits in quantization, which is approximately equal to seven in our system. N is 2400 in our case. So, the theoretical bound σ_ϕ is 0.0075 degrees.

TABLE II
ESTIMATED PHASE FOR 7 CONSECUTIVE BLOCKS

Sample Block	Estimated Phase	Estimated Error	
1	56.90061	0.02108	0.0059%
2	56.88307	0.00355	0.0010%
3	56.89102	0.01146	0.0032%
4	56.87873	-0.00080	-0.0002%
5	56.87794	-0.00160	-0.0004%
6	56.89712	0.01759	0.0049%
7	56.87957	0.00006	0.00001%

Phase estimates (in degrees) for the IADC versus the phase determined by post-processing analysis of sample block 1. This analysis determined that the actual phase was 56.8795 degrees, measured by the FPGA hardware implementation to an accuracy of better than 0.01 degree. Estimate errors in percentages are taken as a proportion of the measured phase error out of 360 degrees.

C. Differential Channel Phase

The second experiment used to validate the phase estimator involved a phase comparison of signals arriving simultaneously on both I and Q channels. In this experiment, the phase is determined using the iterative method and the results are subtracted from one another to determine the phase difference.

As shown in Figure 9, a single signal from the arbitrary waveform generator is split into I and Q channels. When equal length (2m) cables are used for both channels, only minor phase differences are expected. While there are various reasons that the phase might be different for the two channels (e.g. timing differences between the two ADCs), the time-varying phase difference, if performed accurately, shows path integrated differences in electrical length between the two channels that feed into the ADCs. Because the cable electrical length is due in part to their physical length, and their physical length can be changed by changing the cable temperatures, it is expected that such a measurement would show path integrated temperature differences between the two cables. For this experimental case, the two cables instances are the same length, and co-located. Thus, it is expected that the effect of temperature variations in the lab environment would effect both cables equally, hence showing no change in the phase difference between the two cables over time.

For this experiment, similar to the single channel phase measurement experiment, two shadow FIFOs were used to store copies of data for post processing. Actual (determined from post-processing) and FPGA-estimated phases for a split 62MHz signal input into both I and Q ADC channels are shown for 7

TABLE III
ESTIMATED DIFFERENTIAL PHASE FOR 7 CONSECUTIVE BLOCKS

Sample Block	True Phase	Estimated Phase	Difference Error	
1	42.56018	42.56093	0.00074	0.00021%
2	42.56095	42.56093	-0.00002	-0.00001%
3	42.56646	42.56967	0.00321	0.00089%
4	42.55202	42.55219	0.00017	0.00005%
5	42.54841	42.54869	0.00029	0.00008%
6	42.53650	42.53645	-0.00006	-0.00001%
7	42.54269	42.54170	-0.00097	-0.00052%

Phase differences (in degrees) between two channels over seven consecutive blocks. Errors in percentages are taken as a proportion out of 360 degrees. The column labeled *True Phase* is calculated through data analysis of the input samples in post-processing.

blocks of data in Table III. It can be seen that only a very small error is present, likely due to round off and quantization errors. It can be seen that the phase difference between the post-processed and FPGA processed data is constrained within 0.0032 degree (within the theoretical uncertainty), an indication that the FPGA phase measurement errors are systematic, and therefore cancel one another when calculating phase differences.

D. Real-Time Differential Phase Detection

In a second differential phase measurement experiment where data was collected and analyzed by the FPGA system in real-time, both FIFOs were flushed every second and a new sequence of samples were fetched from the ADCs. The differential phases of 8 consecutive blocks were calculated and the mean value was taken as the differential phase in that second and then reported through a simple serial-bus connection. The phase difference between the two channels was calculated by the FPGA at one second time intervals and reported to the serial bus at one second time intervals. A computer monitored this serial port and recorded the data shown in Figs 11 and 12.

An initial experiment experiment was performed using two 2m cables for I and Q channel inputs. Real-time phase differences measured over 50 minutes are shown in Fig. 11. A second experiment was performed using an input cable of 2m and one of 30m. Real-time phase differences measured over 120 minutes are shown in Fig. 12. For this second experiment, the temperature of the longer cable was measured during the time period of the experiment using an Agilent 34970A data acquisition device. The temperature curve is plotted in the lower curve in Fig. 12.

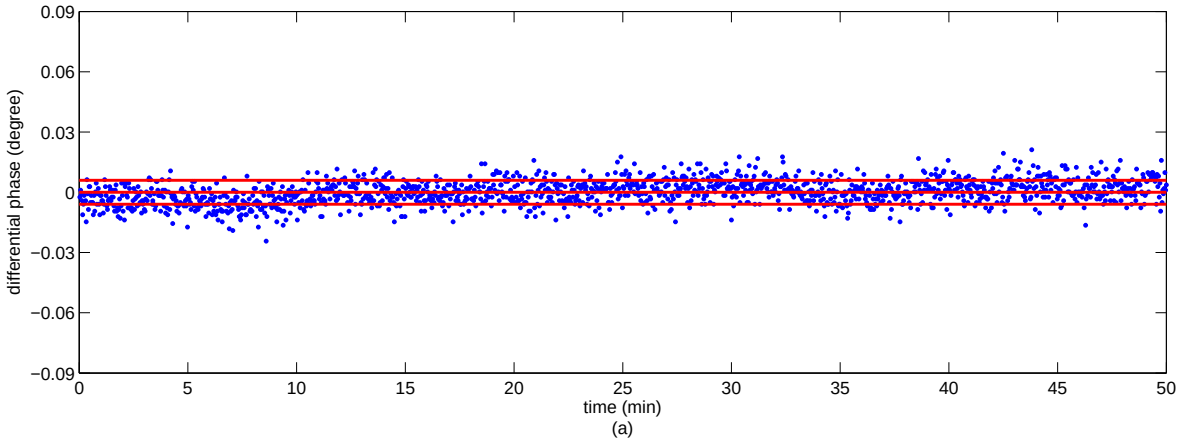

Fig. 11. Real-time differential phase using a split signal routed through two 2 m cables into the inputs of the I and Q ADC channels.

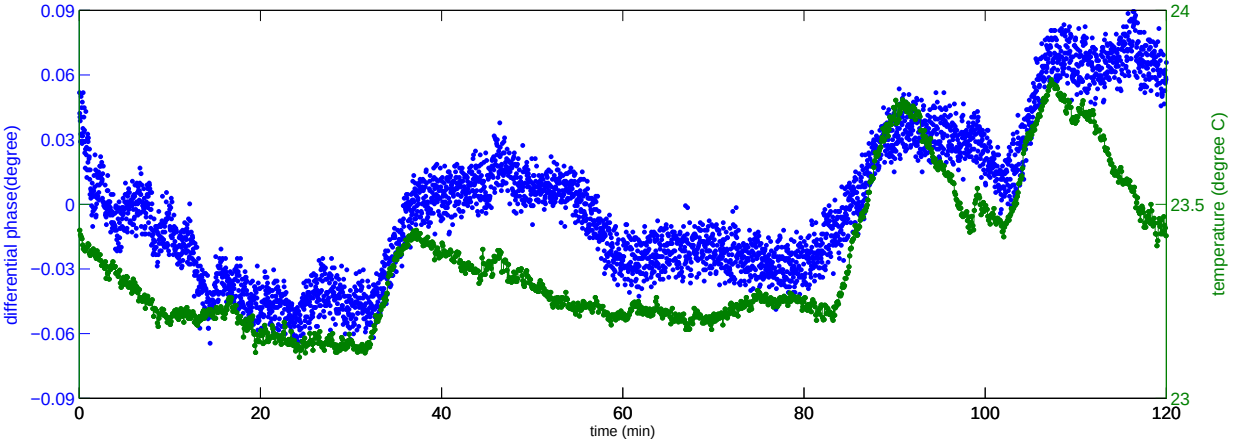

Fig. 12. The real-time differential phase for a split signal sent through a 2 m and a 30 m cable into the inputs of the I and Q ADC channels (upper curve). The lower curve shows the temperature variation of the longer cable during the data collection time period.

Since the length of copper cable used in the experiment changes due to thermal expansion, the electrical path length difference (measured through the phase difference) also changes as a result of changing cable length. When the cables are the same length, the fluctuation of room temperature has roughly the same impact on both cables, so little phase difference is expected over time. This is experimentally verified by the results plotted in Fig. 11. When the cable lengths are different, however, as in the second experiment, room temperature fluctuations have an imbalanced impact on the two signals because a temperature change in the laboratory will disproportionately affect the path integrated electrical length of the longer cable more compared to the shorter one. This is verified by the results shown in Fig. 12, where the phase differences roughly follow the temperature changes also plotted in the figure.

Note, in Fig. 11, horizontal lines show the mean phase difference (zero degrees) and a measured standard

deviation of 0.006 degrees on either side of this mean. This measured standard deviation is consistent with the value derived from (18) which indicates that the variation in phase measurements is due to the signal SNR and the number of samples collected. Note too, that the fluctuations in differential phase shown in Fig. 12 are larger than this standard error. This finding indicates that the large-scale phase fluctuations are real and hence likely due to effects (such as temperature fluctuations) occurring in the physical system outside of the FPGA hardware estimation algorithm. These results are entirely consistent with the explanation given in the preceding paragraph.

E. Temperature Tracking Using Differential Phase Detection

To complete this demonstration and to investigate how temperature impacts phase changes in copper cable, the following experiment was performed. We placed the 30m cable in a thermally-isolated container and used a fan to maintain air circulation near the cable so that temperature would remain uniform within the container around the cable. A cup of boiling water was then placed in the box so air inside the box would gradually heat up. The differential phase between the two input channels (similar to the experiment in Section VI-D) and the temperature were simultaneously measured. The measured differential phase and temperature curves are plotted in Fig. 13(a). The bottom line is the differential phase curve and the top line is the temperature curve measured over 90 minutes. Supposing linearity between the temperature and the phase, one should expect that the phase curve and the temperature curve match each other during the measuring period by properly scaling these two curves. However, the temperature changes faster than the phase in the experiment: increasing faster when heating up and decreasing faster when cooling down. This non-linearity is caused by the thermal inertia of the copper: thermal expansion could not instantly follow the temperature change measured in the environment. Hence, a lag exists between the thermally-sensed temperature and the temperature sensed by the phase measurement technique. Fig. 13(b) shows a plot of temperature versus phase and well illustrates this phenomenon: two phase values are observed for a single temperature (one each during the heating and cooling phases). This will be further discussed momentarily.

In a second temperature-related experiment, we place ice instead of hot water in the box. The resulting phase and temperature curves are plotted in Fig. 14(a). In this figure, a good linear correlation between phase and temperature can be observed.

To interpret these results further, the relationship between the cable length L and temperature T is modeled by

$$L = L_0(1 + \rho T) \tag{20}$$

where, ρ is the linear thermal expansion coefficient for copper and the electrical length ϕ is given by

$$\phi = \frac{2\pi}{\lambda} L. \tag{21}$$

Hence, the phase change $\Delta\phi$ is determined by the temperature change ΔT, given by

$$\Delta\phi = \frac{2\pi}{\lambda} \rho \Delta T. \tag{22}$$

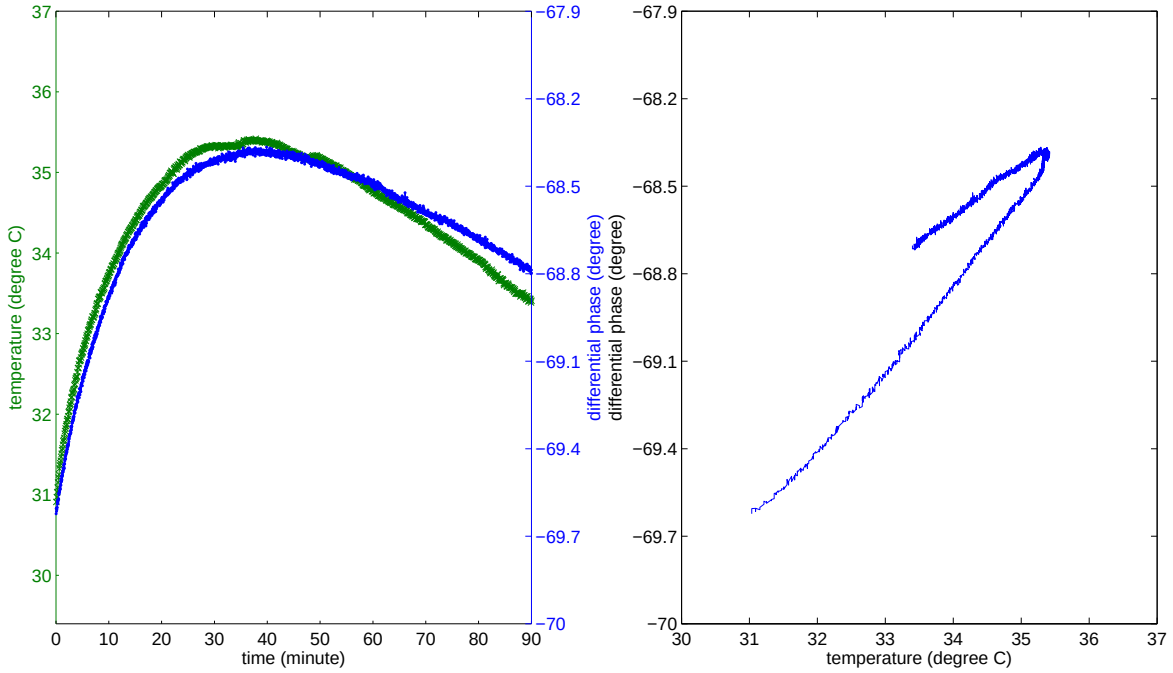

Fig. 13. (a) Differential phase (bottom curve on left) and temperature (top curve on left) under heating conditions. (b) A plot of phase versus temperature.

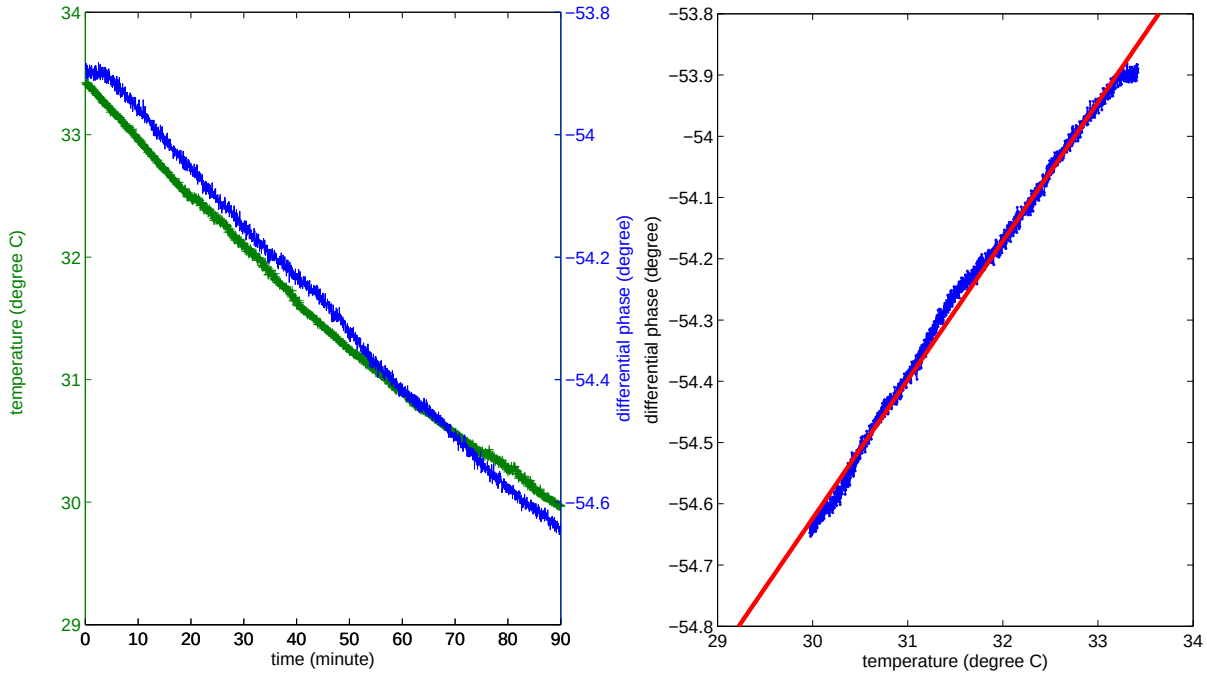

Fig. 14. (a) Differential phase (blue curve - top curve on left) and temperature (green curve - bottom curve on left) under cooling conditions. (b) A plot of phase versus temperature and a linear curve fit.

For a 62 MHz signal and 30m cable length, the thermal expansion coefficient is $62.8 \times 10^{-6}/°\text{C}$. The theoretical value is $16.7 \times 10^{-6}/°\text{C}$ at room temperature. It is expected that the difference between these two coefficients derives from the simplicity of the model which does not take into account the effect of the cable connectors (for instance). While a more sophisticated model could be constructed to demonstrate the relation between phase and temperature, this was not the focus of our work.

Assuming a linear relationship between temperature and phase, the differential phase can be modeled with respect to temperature as

$$\phi_{\text{diff}} = aT + b. \tag{23}$$

where a is the coefficient of expansion and b is a constant related to the differential path length of the two cables. Using a least squares method, the model can be fitted to the data, as shown in Fig. 14(b) and the coefficients determined ($a = 0.22644$ and $b = -61.417$). With the thermal expansion coefficient, we can predict the actual cable temperature from the phase measurements shown in Fig. 13 using (23). The relative temperature is obtained and the curve of actual (from the phase differences) and thermally-measured cable temperatures are plotted in Fig. 15. The reference line with slope 1 is also plotted in the same figure. One can observe that there are two actual cable temperatures for one measured temperature: one for cooling and another for heating. When the cable is heated, the actual temperature of the cable is lower than the measured one because of a lag between the thermally-measured temperature and the path integrated temperature derived from the phase measurements. For the case when the environment is cooling, this relationship is reversed and the phase-difference measured temperature is higher than the one measured by the temperature sensor.

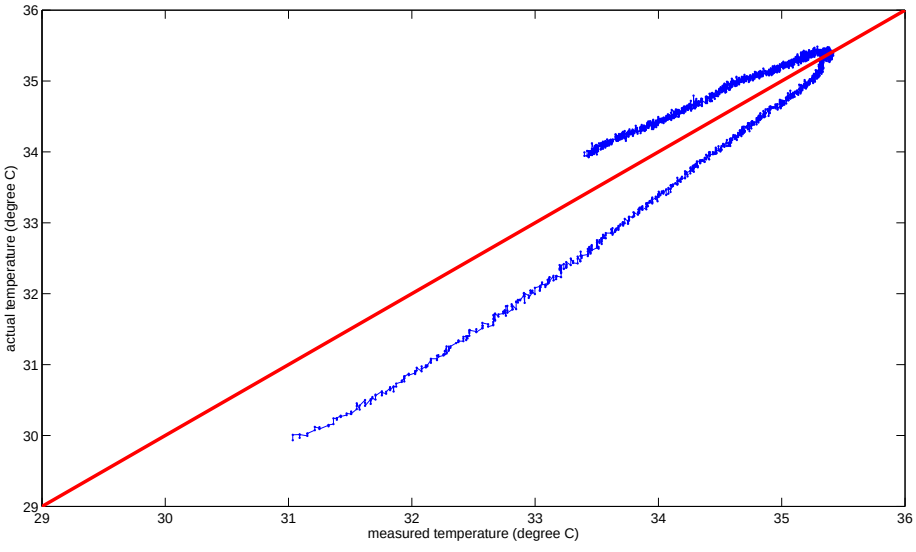

Fig. 15. Cable temperature determined through the integrated path length difference method vs. the temperature measured with a traditional thermal sensor. A diagonal line with a slope of one is included for reference.

VII. Conclusion and Future Work

For many spaceborne systems, the on-board rapid calculation of input signal phase differences can provide important diagnostic information. The presence of high-performance adaptive hardware in the signal processing data path provides an opportunity to quickly and efficiently evaluate signal phase and identify phase drift. In this project, a phase estimation circuit has been implemented in FPGA hardware and experimentally tested. We demonstrate the approach using a new state-of-the-art data acquisition system developed for the NASA Surface Water Ocean Topography (SWOT) project. Phase calculations with maximal accuracy for a given SNR and number of samples were determined using our adaptive approach. A series of measurements were conducted to demonstrate the utility of this algorithm and to verify its accuracy. With the real-time phase detection system, it has been demonstrated that minor temperature and phase changes can be measured in real-time with this system. In the future, we will consider the reliability of our FPGA circuits in the presence of single event upsets (SEUs). Circuit redundancy may be necessary to ensure continued accurate circuit operation in the event of SEUs.

Acknowledgment

This research was funded by NASA's Earth Science Technology office under a grant from the Advanced Component Technology program (grant #ACT-08-0048). The authors wish to thank National Semiconductor, Xilinx Corporation, and MS Kennedy for their donation of circuit components. The assistance of Akilesh Krishnamurthy is also appreciated.

References

[1] P. Siqueira, R. R. Ahmed, J. Wirth, and A. Bachmann, "Variable Precision Two-Channel Phase, Amplitude, and Timing Measurements for Radar Interferometry and Polarimetry," *IEEE Transaction on Microwave Theory and Techniques*, vol. 55, no. 10, pp. 2248–2256, 2007.

[2] "Earth Science and Applications from Space: National Imperatives for the Next Decade and Beyond," National Research Council, Tech. Rep., 2007.

[3] J. Curlander and R. McDonough, *Synthetic aperture radar - Systems and signal processing*. John Wiley and Sons, 1991.

[4] P. Rosen, S. Hensley, I. Joughin, F. Li, S. Madsen, E. Rodriguez, and R. Goldstein, "Synthetic Aperture Radar Interferometry," in *Proceedings of the IEEE*, vol. 88, no. 3, pp. 333–382.

[5] B. Tapley, S. Bettadpur, M. Watkins, and C. Reigber, "The Gravity Recovery and Climate Experiment: Mission Overview and Early Results," *Geophysical Research Letters*, vol. 31, no. 9, May 2004.

[6] A. Guntoro, P. Zipf, O. Soffke, H. Klingbeil, M. Kumm, and M. Glesner, "Implementation of Realtime and Highspeed Phase Detector on FPGA," in *Reconfigurable Computing: Architectures and Applications*, ser. Lecture Notes in Computer Science, K. Bertels, J. Cardoso, and S. Vassiliadis, Eds. Springer Berlin / Heidelberg, 2006, vol. 3985, pp. 1–11.

[7] Z. Wang, L. Mao, and R. Liu, "High-Accuracy Amplitude and Phase Measurements for Low-Level RF Systems," *IEEE Transactions on Instrumentation and Measurement*, vol. PP, no. 99, pp. 1–10, 2012.

[8] L. Esteban, M. Sanchez, J. A. Lopez, O. Nieto-Taladriz, P. Pedreira, and P. Acedo, "Development of efficient FPGA-based phase meters for IR-interferometers. Optimizations for Multi-Channel Interferometers," in *Real Time Conference (RT), 2010 17th IEEE-NPSS*, May 2010, pp. 1–7.

[9] J.-L. Bertaux, D. Nevejans, O. Korablev, E. Villard, E. Quémerais, E. Neefs, F. Montmessin, F. Leblanc, J. Dubois, E. Dimarellis, A. Hauchecorne, F. Lefèvre, P. Rannou, J. Chaufray, M. Cabane, G. Cernogora, G. Souchon, F. Semelin, A. Reberac, E. V. Ransbeek, S. Berkenbosch, R. Clairquin, C. Muller, F. Forget, F. Hourdin, O. Talagrand, A. Rodin, A. Fedorova, A. Stepanov, I. Vinogradov, A. Kiselev, Y. Kalinnikov, G. Durry, B. Sandel, A. Stern, and J. Gérard, "SPICAV on Venus Express: Three spectrometers to study the global structure and composition of the Venus atmosphere," *Planetary and Space Science*, vol. 55, no. 12, pp. 1673 – 1700, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0032063307000074

[10] B. Osterloh, H. Michalik, S. Habinc, and B. Fiethe, "Dynamic Partial Reconfiguration in Space Applications," in *NASA/ESA Conference on Adaptive Hardware and Systems, 2009. AHS 2009.*, 29 2009-Aug. 1 2009, pp. 336 –343.

[11] Z. Ruan, Y. Han, H. Cai, S. Jin, and J. Han, "A Dynamically Partial-reconfigurable FPGA-based Architecture for Data Processing on Space Solar Telescope," in *International Symposium on Industrial Embedded Systems, 2007. SIES '07.*, July 2007, pp. 194 –199.

[12] B. Osterloh, H. Michalik, B. Fiethe, and F. Bubenhagen, "Enhancements of Reconfigurable System-on-Chip Data Processing Units for Space Application," in *Second NASA/ESA Conference on Adaptive Hardware and Systems, 2007. AHS 2007.*, Aug. 2007, pp. 258 –262.

[13] P. Siqueira, R. Tessier, A. Swochak, K. Fu, D. Esteban-Fernandez, and B. Heavey, "A Single-Stage, Two-Channel Ka-band to Digital, Thermal Compensating Receiver for SWOT," in *NASA Earth Science Technology Forum*, Arlington, VA, 2010.

[14] P. Siqueira, K. Srinivasan, E. Insanic, and R. Ahmed, "A Cross-Track Ku-Band Interferometer for Topographic and Volumetric Depth Measurements," in *IEEE Aerospace Conference Proceedings*, March 2007, pp. 2214–2221.

[15] P. Siqueira, R. Tessier, T. Hartley, B. Heavey, D. Esteban-Fernandez, and M. Nakashima, "A Ka-band to Baseband RF Testbed for the SWOT mission," in *NASA Earth Science Technology Forum*, Pasadena, CA, 2011.

[16] V. Vijayendra, P. Siqueira, H. Chandrikakutty, A. Krishnamurthy, and R. Tessier, "Real-Time Estimates of Differential Signal Phase for Spaceborne Systems Using FPGAs," in *Proceedings of the NASA/ESA Conference on Adaptive Hardware and Systems*, San Diego, CA, June 2011.

[17] *Floating Point Operator v5.0, DS 335 data sheet*, Xilinx Corportation, June 2009.

[18] *Cordic v4.0, DS 249 data sheet*, Xilinx Corportation, April 2009.